Institute of Software Engineering

Higher Diploma in Software Engineering

Midterm Examination (Model Paper)

Python for Data Science & AI

Duration: 1 Hour 20 minutes (80 Minutes)

This paper has 17 pages including this page and any appendices.

Instruction to students:

- The total marks for this paper is **100**.
- You should mark your answers on the given physical paper (e.g., **underline the correct answer(s)**) using a **blue or black pen**.
- Write your **full name, student id** and **email address** on the **front page** of the given paper (e.g., top right-hand side).
- You should provide answers to all the given questions. Most of the questions have only one correct answer.
- However, some questions may have multiple correct answers. In such cases, you must mark all the correct answers to receive the full mark, as no partial marks will be awarded. These questions can be identified by the phrase "**Select all that apply**".
- You will not be allowed to use notebooks, laptops, mobile phones, or the internet during the exam.

1.) What is the key difference between the set.add() and set.update() methods in Python?

A) .add() can add multiple items, while .update() can only add one.

B) .add() adds a single element, while .update() adds all elements from an iterable.

C) .add() is used for adding lists, while .update() is used for adding tuples.

D) .add() returns a new set, while .update() modifies the set in-place.

E) They are identical and can be used interchangeably.

2.) What is the result of the following code?

```python
my_tuple = (10, 20, [30, 40])

my_tuple[2].append(50)

print(my_tuple)
```

A) (10, 20, [30, 40]) B) (10, 20, [30, 40, 50])

C) A TypeError is raised because tuples are immutable.

D) A ValueError is raised. E) A SyntaxError is raised.

3.) What is the primary difference between accessing a dictionary value with my_dict[key] versus my_dict.get(key)?

A) .get(key) is faster than [key].

B) [key] can set a default value if the key is missing; .get(key) cannot.

C) [key] raises a KeyError if the key is missing, while .get(key) returns None (or a specified default).

D) .get(key) raises a ValueError if the key is missing, while [key] returns None.

E) [key] works for string keys, while .get(key) only works for integer keys.

4.) What is the content of `my_dict` after the following code is executed?

```python
my_dict = {

    'apple': 5,

    'banana': 10,

    'apple': 12

}

print(my_dict)
```

A) {'apple': 5, 'banana': 10, 'apple': 12}

B) {'apple': 5, 'banana': 10}

C) {'banana': 10, 'apple': 12}

D) {'apple': 12, 'banana': 10}

E) A SyntaxError is raised because 'apple' is repeated.

5.) What will be the output of the following code?

```python
my_dict = {'a': 1, 'b': 2}
values = my_dict.values()
my_dict['c'] = 3
print(list(values))
```

A) [1, 2]

B) [1, 2, 3]

C) [('a', 1), ('b', 2), ('c', 3)]

D) dict_values([1, 2])

E) An error is raised because the dictionary was modified.

6.) What will be the output of the following code?

```python
score = 88
grade = ''
if score >= 90:
    grade = 'A'
elif score >= 80:
    grade = 'B'
elif score >= 70:
    grade = 'C'
else:
    grade = 'F'
print(grade)
```

A) A B) B C) C D) F E) No output

7.) What does the following ternary operator expression print?

```python
x = 10
y = 20
result = "Red" if x > y else "Blue" if x == y else "Green"
print(result)
```

A) Red B) Blue C) Green D) RedGreen E) SyntaxError

8.) What is the output of the following match statement?

```
Python
code = (404, "Not Found")
match code:
    case (200, _):
        print("OK")
    case (404 | 403, msg):
        print(f"Error: {msg}")
    case _:
        print("Unknown")
```

A) OK B) Error: 404 C) Error: Not Found D) Unknown E) SyntaxError

9.) What is the output of the following match statement?

```
Python
val = [10, "High", 5]
match val:
    case [x, "High", y] if x > y:
        print(x - y)
    case [x, _, y] if x < y:
        print(x + y)
    case [10, "High", 5]:
        print(0)
    case _:
        print(-1)
```

A) 5 B) 15 C) 0 D) -1 E) SyntaxError

10.) What is the output of the following code snippet?

```python
my_list = []
my_dict = {}
if my_list:
    print("List")
elif my_dict:
    print("Dict")
else:
    print("Empty")
```

A) List B) Dict C) Empty D) ListDict E) No output

11.) What is the output of the following code?

```python
n = 10
while n > 0:
    n -= 2
    if n == 4:
        break
    print(n, end=" ")
else:
    print("Done")
```

A) 8 6 B) 8 6 Done C) 8 6 4 D) 8 6 4 Done E) 10 8 6

12.) What is the output of the following code?

```
Python
total = 0
for i in range(10, 0, -2):
    total += 1
print(total)
```

A) 0 B) 4 C) 5 D) 6 E) 10

13.) What does the following nested loop print?

```
Python
result = ""
for i in range(3):
    for j in range(i):
        result += f"({i},{j}) "
print(result)
```

A) (0,0) (1,0) (1,1) (2,0) (2,1) (2,2) B) (1,0) (2,0) (2,1) C) (0,0) (1,0) (2,0)

D) (1,0) (1,1) (2,0) (2,1) E) No output

14.) What is the output of the following code?

```
Python
count = 0
for num in [1, 2, 3, 4, 5]:
    if num % 2 == 0:
        continue
    count += num
print(count)
```

A) 15 B) 6 C) 9 D) 1 E) 3

15.) What is the output of iterating over a dictionary's .items()?

```
Python
data = {'x': 100, 'y': 200}
for item in data.items():
    print(item)
```

A) x
   y

B) 100
   200

C) ('x', 100)
   ('y', 200)

D) x 100
   y 200

E) dict_items([('x', 100), ('y', 200)])

16.) What is the final state of numbers after this loop?

```
Python
numbers = [1, 2, 3, 4, 5, 6]
for i in range(len(numbers)):
    if numbers[i] % 3 == 0:
        numbers.pop(i)
```

A) [1, 2, 4, 5] B) [1, 2, 3, 4, 5, 6] C) [1, 2, 4, 5, 6]

D) [1, 2, 4, 5] E) An IndexError is raised.

17.) What is the output of the following code?

```python
data = {'a': 10, 'b': 20, 'c': 30}

total = 0

for k in data:

    if k == 'b':

        continue

    total += data[k]

print(total)
```

A) 60 B) 30 C) 50 D) 20 E) 40

18.) What is the output of the following code?

```python
count = 5
total = 0
while True:
    total += count
    count -= 1
    if count == 0:
        break
print(total)
```

A) 5 B) 10 C) 15 D) 0 E) This is an infinite loop.

19.) Which statement about function arguments in Python is correct?

A) Keyword arguments must always be provided before positional arguments.

B) A function must have at least one argument.

C) Positional arguments are matched by name, and keyword arguments are matched by order.

D) Once a keyword argument is used in a function call, all subsequent arguments must also be keyword arguments.

E) Default values can only be provided for positional arguments.

20.) What is the output of the following code?

```python
def add_item(item, my_list=[]):
    my_list.append(item)
    return my_list

print(add_item(1))
print(add_item(2))
```

A) [1]
[2]

B) [1]
[1, 2]

C) [1]
None

D) None
None

E) A TypeError is raised.

21.) What will be the output of the following code?

```python
def calculate(a, b, c=10, d=5):
    return (a + b) * c - d

print(calculate(3, 7, d=20))
```

A) 80 B) 15 C) 100 D) 85 E) A TypeError is raised.

22.) What is the output of the following code?

```python
x = 10
def my_func():
    global x
    x = 20
    print(f"Inner: {x}", end=" ")

my_func()
print(f"Outer: {x}")
```

A) Inner: 20 Outer: 10 B) Inner: 20 Outer: 20 C) Inner: 10 Outer: 10

D) Inner: 10 Outer: 20 E) A **NameError** is raised.

23.) What will the following function return when called?

```python
def get_stats(numbers):
    return min(numbers), max(numbers), sum(numbers) / len(numbers)

stats = get_stats([1, 9, 10])
print(type(stats))
```

A) <class 'list'> B) <class 'int'> C) <class 'tuple'>

D) <class 'dict'> E) <class 'float'>

24.) Given the function def test(a, b=5, c=10):, which of the following calls are valid? Select all that apply.

A) test(1, 2, 3) B) test(a=1, 2, 3) C) test(1, c=20)

D) test() E) test(5, b=10, c=15)

25.) What is the main purpose of the **kwargs syntax in a Python function definition?

A) It allows the function to accept any number of positional arguments.

B) It requires the caller to pass a dictionary named kwargs.

C) It collects all keyword arguments not explicitly named in the signature into a dictionary.

D) It allows the function to return multiple values.

E) It unpacks a dictionary into positional arguments.

26.) What will the following code output?

Python
```python
def summarize(*args):
    if len(args) == 0:
        return 0
    return sum(args) / len(args)

print(summarize(10, 20, 30))
```

A) 60 B) 20.0 C) (10, 20, 30) D) 3 E) A TypeError is raised.

27.) What will be the output of the following code?

```python
def process_data(**kwargs):
    status = kwargs.pop('status', 'Pending')
    return f"Status: {status}, Remaining: {len(kwargs)}"

print(process_data(id=101, user='admin', status='Complete'))
```

A) Status: Complete, Remaining: 2

B) Status: Pending, Remaining: 2

C) Status: Complete, Remaining: 3

D) Status: Pending, Remaining: 3

E) A **KeyError** is raised.

28.) Which of the following function signatures is syntactically valid in Python? Select all that apply.

A) def f(a, *args, b=10, **kwargs):

B) def f(a, b=10, *args, **kwargs):

C) def f(a, *args, **kwargs, b=10):

D) def f(*args, a, b=10, **kwargs):

E) def f(a, b=10, **kwargs, *args):

29.) See the code below. Which option(s) correctly call the function and produce the specified output? Select all that apply.

```python
def setup(name, *args, **config):
    print(f"Name: {name}")
    print(f"Args: {len(args)}")
    print(f"Config: {'debug' in config}")

# Desired Output:
# Name: Main
# Args: 2
# Config: True
```

A) setup("Main", 1, 2, debug=True) B) setup(name="Main", 1, 2, debug=True)

C) setup("Main", (1, 2), debug=True) D) setup("Main", 1, 2, config={'debug': True})

E) setup("Main", 1, 2, 3, debug=False)

30.) What is the output of the following code?

```python
def calculate(x, y, z):
    return x * y + z


data = [10, 5]
config = {'z': 2}
print(calculate(*data, **config))
```

A) 17 B) 52 C) 107 D) 25 E) A TypeError is raised.

31.) What is the final value of new_list?

```python
my_list = [1, 2, 3, 4, 5]
new_list = [x * 10 for x in my_list]
print(new_list)
```

A) [1, 2, 3, 4, 5] B) [10, 20, 30, 40, 50] C) [10]

D) [50] E) A NameError is raised.

32.) What is the value of filtered_list after this list comprehension?

```python
data = "The 1 quick 2 brown 3 fox"
filtered_list = [w.upper() for w in data.split() if w.isalpha()]
print(filtered_list)
```

A) ['THE', '1', 'QUICK', '2', 'BROWN', '3', 'FOX']

B) ['1', '2', '3']

C) ['The', 'quick', 'brown', 'fox']

D) ['THE', 'QUICK', 'BROWN', 'FOX']

E) ['T', 'H', 'E', 'Q', 'U', 'I', 'C', 'K', ...]

33.) What is the output of the following nested list comprehension?

```python
matrix = [[1, 2], [3, 4]]
flat_list = [num * 2 for row in matrix for num in row]
print(flat_list)
```

A) [[2, 4], [6, 8]]

B) [1, 2, 3, 4]

C) [2, 4, 6, 8]

D) [2, 6]

E) [[2, 4], [3, 4]]

34.) Which of the following statements correctly describes a Python lambda function?

A) It can contain multiple expressions, but only one statement.

B) It is defined using the def keyword, just like a regular function.

C) Its body is restricted to a single expression and it returns the result of that expression.

D) It cannot accept any arguments.

E) It must be assigned to a variable to be used.

35.) What is the output of the following code?

```
Python
words = ["apple", "banana", "cherry"]
result = list(map(lambda x: x[0], words))
print(result)
```

A) ['a', 'b', 'c'] B) ['apple', 'banana', 'cherry'] C) ['a', 'p', 'p', 'l', 'e', ...]

D) ['apple'] E) A TypeError is raised.

36.) What is the output of the following code?

```
Python
numbers = [1, 2, 3, 4, 5, 6]
result = list(filter(lambda x: x % 2 == 0, numbers))
print(result)
```

A) [1, 3, 5] B) [2, 4, 6] C) [True, False, True, False, True, False]

D) [False, True, False, True, False, True] E) A ValueError is raised.

37.) What is the output of using map() with multiple iterables?

```
Python
list_a = [1, 2, 3]
list_b = [10, 20, 30, 40]
result = list(map(lambda x, y: x + y, list_a, list_b))
print(result)
```

A) [11, 22, 33] B) [11, 22, 33, 40] C) [11, 22, 33, 41]

D) A ValueError is raised because the lists are different lengths.

E) A TypeError is raised.

38.) What is the direct output of printing the filter_obj variable?

```
Python
scores = [10, 50, 90]
filter_obj = filter(lambda x: x > 40, scores)
print(filter_obj)
```

A) [50, 90] B) (50, 90) C) [False, True, True]

D) A string showing the type and memory address of a filter object.

E) An StopIteration error is raised.

39.) What is the value of total after executing this chain?

```
Python
data = [1, 2, 3, 4, 5]
mapped = map(lambda x: x * x, data)
filtered = filter(lambda x: x > 10, mapped)
total = sum(filtered)
print(total)
```

A) 15 B) 55 C) 41 D) 50 E) 0

40.) From a practical standpoint, which is the most accurate comparison between map and a list comprehension?

A) map is always faster than a list comprehension.

B) List comprehensions are generally considered more "Pythonic" and readable for simple transformations than map.

C) map can be used with if conditions, while list comprehensions cannot.

D) List comprehensions return an iterator, while map returns a fully-formed list.

E) They are identical in every way, and the choice is purely personal preference.

Answers

1. B)
2. B)
3. C)
4. D)
5. B)
6. B)
7. C)
8. C)
9. A)
10. C)
11. A)
12. C)
13. B)
14. C)
15. C)
16. E)
17. E)
18. C)
19. D)
20. B)
21. A)
22. B)
23. C)
24. A, C, E
25. C)
26. B)
27. A)
28. A, B, D
29. A)
30. B)

31. B)

32. D)

33. C)

34. C)

35. A)

36. B)

37. A)

38. D)

39. C)

40. B)